A Privacy-Preserving Protocol for Network-Neutral Caching in ISP Networks

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 5, 2019, accepted October 24, 2019, date of publication October 30, 2019, date of current version November 14, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2950593

A Privacy-Preserving Protocol for Network-Neutral Caching in ISP Networks

DAVIDE ANDREOLETTI[1,2], OMRAN AYOUB[2], CRISTINA ROTTONDI[3], SILVIA GIORDANO[1], GIACOMO VERTICALE[2], AND MASSIMO TORNATORE[2]

[1]Networking Laboratory, University of Applied Sciences of Southern Switzerland, 6928 Manno, Switzerland
[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133 Milano, Italy
[3]Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Turin, Italy

Corresponding author: Davide Andreoletti (davide.andreoletti@supsi.ch)

The authors D. Andreoletti and S. Giordano were supported by the Swiss National Science Foundation (SNSF) via the CHIST-ERA project UPRISE-IoT. The work of G. Verticale was supported in part by the project ATMOSPHERE, in part by the Brazilian Ministry of Science, Technology and Innovation (Project 51119-MCTI/RNP 4th Coordinated Call), and in part by the European Commission under the Cooperation Programme, Horizon 2020.

ABSTRACT By performing in-network caching, Internet Service Providers (ISPs) allow Content Providers (CPs) to serve contents from locations closer to users. In this way, the pressure of content delivery on ISPs' network is alleviated, and the users' Quality-of-Experience (QoE) improved. Due to its impact on QoE, caching has been recently considered as a form of traffic prioritization in the debate on Network Neutrality (NN). A possible approach to perform NN-compliant caching consists in assigning the same portion of cache storage to all the CPs. However, this static subdivision does not consider the different popularities of the CPs' contents and is therefore inefficient. Alternatively, the cache can be subdivided among the CPs proportionally to the popularity of their contents. However, CPs consider this information private and are reluctant to disclose it. In this work, we propose a protocol to perform a *popularity-driven* subdivision of the caches' storage in a privacy-preserving and network-neutral fashion. The protocol is based on the Shamir Secret Sharing (SSS) scheme and is designed to ensure a NN-compliant subdivision of the caches while preserving the privacy of both CPs and ISP (i.e., contents' popularity and caches' size are not disclosed). Through dynamic simulation, we show that the popularity-driven cache subdivision (enforced by using our protocol) outperforms several baseline approaches in terms of overall network Resource Occupation (RO) and caching Hit-Ratios. Thanks to our numerical results, we observe that the frequency of execution of the protocol has a significant impact on the RO, and that the ISP can tune this frequency to minimize its RO while introducing an acceptable data overhead. Because of this tuning, several CPs may experience a loss with respect to the hit-ratio that they would obtain by independently choosing the frequency of execution. This loss is very limited, and the employment of the protocol is therefore beneficial to all the involved parties, especially since, by using it, CPs are guaranteed that the ISP behaves in a network-neutral manner.

INDEX TERMS Network neutrality, privacy, content providers, privacy-preserving caching.

I. INTRODUCTION

Online video streaming, especially Video-on-Demand (VoD), has been a main driving force for the recent escalation in the overall Internet traffic, both for fixed and mobile users. Cisco predicts that, by 2022, 82% of the total Internet traffic will be generated by the distribution of video contents [1], which are owned by over-the-top Content Providers (CPs) and distributed to users connected thanks to an Internet Service Provider (ISP).

The associate editor coordinating the review of this manuscript and approving it for publication was Donghyun Kim.

To cope with VoD traffic growth, ISPs exploit *network caching* to keep network-resource occupation low and to provide users with improved QoE (achieved by lowering retrieval latency and congestion probability [2]). Using caching, portions of CPs' content catalogues (mainly popular contents) can be served from locations closer to end-users (i.e., the caches). Therefore, by establishing a subdivision of the available cache storage capacity among the CPs, the ISP can significantly affect the users' QoE. Hence, caching can be regarded as a form of discriminatory traffic prioritization and it has recently emerged in the debate on Network Neutrality (NN) [3]–[5].

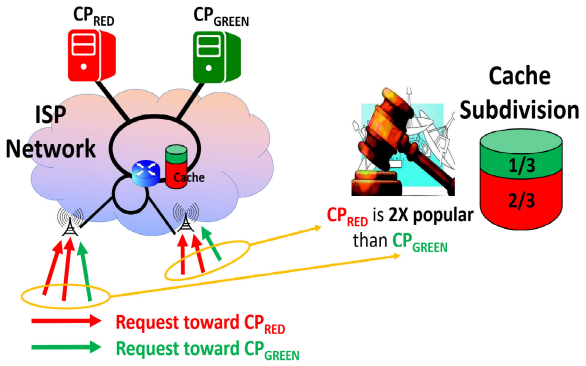


FIGURE 1. High-Level representation of the proposed idea of NN-compliant caching: CPs are entitled to receive a portion of storage proportional to their popularity.

In brief, NN is the legislative principle according to which ISPs are allowed to prioritize a class of traffic only based on its performance requirements. For example, the traffic generated by a VoIP call can be treated with higher priority with respect to e-mail exchange, but not with respect to another call. As far as caching is concerned, however, all the traffic is generated by the same class of contents (e.g., video) and, to the best of our knowledge, there is no current agreement on the definition of NN-compliant caching. A possible view of NN-compliant caching [3], [5] requires the ISP to reserve CPs portions of storage proportional to their contents' popularity. A high-level representation of this concept is depicted in Fig. 1. An example of this type of subdivision, that we refer to as *popularity-driven*, is the following: given a cache server that can store an average number of 1000 contents, a CP that owns 500 out the 1000 most requested contents from the users of the ISP is assigned 50% of the available cache storage. Consistently with our previous works [4], [5], we consider the popularity-driven subdivision to be both NN-compliant and effective. In fact, it is NN-compliant since it guarantees the CPs a neutral treatment (because the storage is assigned only based on their attractiveness and not on arbitrary forms of agreement with the ISP), but also effective for the ISP, that experiences the highest reduction of its network resource occupation when the most popular contents are served directly from its area. For a better understanding, we refer the reader to Ref. [4], where we presented a numerical analysis on how caching can be discriminatory towards the CPs.

On the other hand, compliance to such NN principles would require the ISP to obtain information about contents' popularity, which is unlikely as CPs are increasingly encrypting their contents to protect users' privacy. To cope with this issue, we propose a privacy-preserving protocol by which the ISP can divide its cache storage proportionally to the popularities of CPs' contents. More specifically, the contributions of our work are listed in the following.

- We propose a protocol based on the Shamir Secret Sharing (SSS) scheme that is designed to protect both CPs' and ISP's privacy requirements, as ISP and CPs are not required to exchange with each other sensitive information (e.g., the capacities of the caches and the popularities of contents).
- We define an architecture suitable for the execution of the protocol. This architecture is composed of an ISP, several CPs and a regulator authority (RA), which ensures that each CP is assigned a fair amount of storage and that no illicit collusion occurs among CPs and ISP.
- We develop a discrete-event-based simulator for dynamic VoD traffic provisioning, that we use to perform the comparison between the proposed protocol and two baseline approaches, namely the (i) *static* and the (ii) *network-resource-driven* subdivisions.
- We perform extensive simulations and measure the performance of the various approaches in terms of the network *Resource Occupation* (RO), measured by the ISP, and the *Hit-Rate* measured by the ISP and CPs.

The obtained results show that our proposed protocol allows minimizing network RO and maximizing the overall hit-ratio with respect to baseline approaches and, unlike the baselines, it also guarantees a NN-compliant storage subdivision. Finally, we evaluate the overhead the protocol introduces, which may be considered negligible compared to the reduction of RO that the protocol provides.

A. RELATED WORK

1) NETWORK-NEUTRAL CACHING

Existing literature on NN mainly focuses on legislative aspects [6] and technical approaches for monitoring its fulfillment [7], which is widely-recognized as a rather difficult task. Along this line, it is often argued (e.g., in [8]) that NN can be enforced only if network management becomes more transparent to ISPs customers. In this work, we propose a protocol that guarantees a fair and transparent treatment to CPs that apply caching on an ISP network. While research concerning regulations of NN-compliant traffic prioritization is quite extensive and mature [6], NN aspects of caching have been rarely considered. To the best of our knowledge, caching has been considered as a potentially-discriminatory process only in Ref. [3], [4], where possible definitions of NN-compliant caching are proposed. These works, however, do not propose any technical implementation for the enforcement of NN-compliant caching. Ref. [4] observes that the enforcement of NN-compliant caching is prevented by the wide adoption of encryption. In this work, we focus on the technical implementation of a protocol that enables the enforcement of NN-compliant caching. Specifically, we extend our work in [5], in which we designed a protocol that guarantees (i) NN compliance, (ii) scalability (e.g., with respect to the number of CPs) and (iii) privacy (CPs and ISP are not required to exchange sensitive information with each other). The main limitations of the protocol in [5] are that the available cache storage is not fully utilized and all the contents are required to be of the same size. Both these limitations are overcome in the extended version described in this paper.

2) PRIVACY-PRESERVING CACHING

Ref. [9] proposes two different approaches for ISPs to efficiently cache contents in presence of encryption. In the first approach, the ISP infers information about popularity by analyzing the occurrences of pseudonomys associated with the contents and selects the contents to be cached accordingly. In [10] it is observed that this approach does not fully protect privacy, and guidelines are provided to improve it. In [11], the ISP reserves a slice of cache storage and leaves each CP manage independently its slice remotely. To compute the slice of storage to allocate to a CP, several approaches are proposed such as by localizing the geographic distribution of requests in a privacy-preserving fashion [12] or by analyzing the aggregate hit-rates experienced by each CP [11]. In our protocol, however, we compute the subdivision of the storage to be allocated to a CP meeting more challenging privacy requirements (i.e., the ISP is not required to obtain the hit-rate related to each CP) and, differently from [11], we guarantee that the ISP actually divides its cache in a NN-fashion. To achieve this objective, we base our protocol on consolidated cryptographic primitives, namely the Paillier cryptosystem and the SSS scheme (explained in detail in Section II). An in-depth description of schemes constructed over SSS can be found in [13]–[15].

B. PAPER ORGANIZATION

The paper is structured as follows. In Section II we provide background on the building blocks of the proposed protocol. A definition of NN-compliant caching, along with a formal problem statement is presented in Section III. The entities involved in the execution of the proposed protocol, their objectives and the security assumptions are presented in Section IV. Section V describes the proposed protocol. The simulations settings and the event-driven simulator employed to perform the experiments are described in Section VI. The obtained results and the relative discussion are presented in Section VII. Finally, Section VIII concludes the paper.

II. BACKGROUND

A. PAILLIER CRYPTOSYSTEM

Paillier [16] is a type of asymmetric cryptosystem, whose public and private keys are referred in the rest of the paper to as pub_k and $priv_k$, respectively. Paillier has additive homomorphic properties, i.e., the summation of two (or more) cypertexts is the encryption of the summation of the relative plaintexts. For example, given two pairs of plaintexts m_1, m_2 and the relative ciphertexts $c_1 = Enc(m_1, pub_k)$, $c_2 = Enc(m_2, pub_k)$, it holds that $m = Dec(c, priv_k)$, where $m = m_1 + m_2$ and $c = c_1 + c_2$.

B. SHAMIR SECRET SHARING

A (W, T) Shamir Secret Sharing (SSS) [17] is a cryptographic scheme that allows to share a secret s among a set of W participants in such a way that its reconstruction can only be performed by the collusion of any subset of at least T participants. In the rest of the paper, we use the notation $[\![c]\!]_P$ to indicate the share of s assigned to the participant P.

The SSS is based on the principle that any polynomial of degree $T-1$ can be perfectly reconstructed from the knowledge of T points that it intercepts. Let $s \in \mathbb{Z}_q$ be the secret (with q a prime number greater than all the possible secrets) and let $v_1, v_2, \ldots, v_{T-1}$ be the coefficients of the polynomial, which are random integers uniformly distributed in $[0, q-1]$. The participant P receives $[\![c]\!]_P = (x_P, y_P)$, with x_P an integer number (distinct for each participant) and $y_P = s + u_1 x_P + u_2 x_P^2 + \ldots + u_{T-1} x_P^{T-1}$. The reconstruction of s can be performed by means of interpolation algorithms, e.g., the Lagrange interpolation.

C. PROTOCOL BUILDING BLOCKS

During the execution of the protocol, the operations performed over secrets shared with SSS are based on three main atomic operators, namely the *equality-test*, the *comparison* and the *multiplication*. The equality-test (resp., comparison) operator takes as input the shares $[\![s_1]\!]$ and $[\![s_2]\!]$ and returns $[\![1]\!]$ if $s_1 = s_2$ (resp., $s_1 \leq s_2$) and $[\![0]\!]$ otherwise. The multiplication takes as input $[\![s_1]\!]$ and $[\![s_2]\!]$ and returns $[\![s_1 \cdot s_2]\!]$.

As for the equality-test, we employ the equality-test without bit decomposition described in [14]. The equality-test operator serves as the main building block for the implementation of the *aggregate-if-equal* algorithm [13], that takes in input $([\![s_1]\!], [\![v_1]\!])$ and $([\![s_2]\!], [\![v_2]\!])$, i.e., two pairs of (secret,value) in secret shared form and returns $([\![s_1]\!], [\![v_1 + v_2]\!])$ and $([\![s_2]\!], [\![0]\!])$ if $s_1 = s_2$, whereas the pairs are left unchanged otherwise. In our work, we employ another algorithm presented in [13] that efficiently aggregates a sequence of M (secret,value) pairs in secret shared form by recursively applying the *aggregate-if-equal* algorithm for $M \log M$ times.

Our protocol requires to perform several multiplications of secrets. However, the SSS is not homomorphic with respect to the multiplication (i.e., given the shares of two secrets s_1 and s_2, $[\![s_1 \cdot s_2]\!] \neq [\![s_1]\!] \cdot [\![s_2]\!]$). To address this issue, our protocol exploits the multiplication scheme proposed in [18]. This scheme requires the parties involved in the multiplication to share with each other a multiplicative triple $[\![a]\!]$, $[\![b]\!]$, $[\![c]\!]$ such that $a \cdot b = c$. The security of the multiplication scheme described in [18] is based on the assumption that none of the involved parties is able to obtain the secrets a, b, c from the relative shares $[\![a]\!]$, $[\![b]\!]$, $[\![c]\!]$. The shares of the multiplication triple can be pre-computed in a secure manner using the scheme proposed in [19].

III. NN-COMPLIANT CACHING

A. DEFINITION

In Ref. [4] we advocated the inclusion of caching in the current debate about NN and provided guidelines to reach a possible definition of NN-compliant caching. In this work, we consider caching to be network-neutral if the available ISP's cache storage is divided among the CPs proportionally to the popularity of their contents. This definition allows to balance the requirements of NN (i.e., CPs are treated based on an unbiased criterion instead of on arbitrary forms of agreements) and the legitimate interests of the ISP, which is willing to minimize its network RO in return of the monetary

investment done to buy and maintain the caching system [20]. In the next subsection, we formally present the problem of computing a popularity-driven subdivision of the storage and we briefly introduce the protocol proposed to solve it in a privacy-preserving manner.

B. PROBLEM STATEMENT

We consider a scenario where a sequence of N requests $\mathcal{R} = \{r_1, r_2, \ldots, r_N\}$ is issued from the users within the area of an ISP towards a set of K CPs. The ISP owns a caching system composed of several cache servers. The generic n-th cache is characterized by the size of its storage $S^{(n)}_{cache}$, expressed in bytes, and by an average number of contents that it can store, i.e., $N^{(n)}_{cache}$. Assuming that the average size of the contents is $\hat{s}$, it is possible to derive $N^{(n)}_{cache}$ as $N^{(n)}_{cache} = \lfloor \frac{S^{(n)}_{cache}}{\hat{s}} \rfloor$.

Following the definition of NN-compliant caching given in Section III-A, the total storage of the n-th cache (i.e., $S^{(n)}_{cache}$) should be divided among the K CPs proportionally to the popularity of their contents. Specifically, if the n-th cache can store, on average, $N^{(n)}_{cache}$ contents, the k-th CP is entitled to receive a percentage of the total storage proportional to the number of its contents belonging to the $N^{(n)}_{cache}$ most requested contents from the area of the ISP. The portion of storage $\gamma^{(n)}_k$ that the k-th CP is worth receiving is computed as:

$$\gamma^{(n)}_k = \frac{z_k}{\sum_{j=1}^{K} z_j} \cdot S^{(n)}_{cache}, \quad 1 \leq k \leq K \tag{1}$$

where $S^{(n)}_{cache}$ is the size of the n-th cache, while z_j is the number of contents offered by the j-th CP whose popularity rank is below $N^{(n)}_{cache}$ (we recall that the most popular content has rank equal to 0).

To compute $\gamma^{(n)}_k$, $1 \leq k \leq K$, the ISP and the CPs are required to exchange with each other information that are deemed confidential, such as the size of the caches and the popularity of the contents. The protocol that we propose allows to perform this computation in a privacy-preserving manner. In the next Section, we describe the roles and objectives of the entities involved in the execution of the protocol.

IV. ARCHITECTURE

The proposed protocol is executed by three entities, namely an ISP, the CPs and a Regulator Authority (RA). This Section is devoted to the description of the involved parties, their caching objectives, privacy requirements, and security models.

A. INTERNET SERVICE PROVIDER

The ISP provides Internet connectivity to its users and it is the owner of the caching system exploited by the CPs. Concerning the execution of our protocol, it has the following objectives/requirements.

Caching Objectives: the main performance objective of the ISP is the minimization of its overall network resource occupation (RO). RO is defined as the amount of resources occupied to deliver all requests (more specifically, RO is the product of the number of network links traversed by the duration of a request by the bit-rate of the requested content).

Privacy Requirements: ISPs commonly consider confidential the information related to their infrastructure [21]. In this work, we assume that ISP is not willing to disclose the size of its cache servers, as this may provide precious information on its monetary investment [22]. More specifically, the RA and all the K CPs should not learn any information about the size of the n-th cache (i.e., $S^{(n)}_{cache}$). CP_k, $1 \leq k \leq K$ can learn, at most, a lower bound $\gamma^{(n)}_k$, which is obtained as a licit output of the protocol. In addition, the RA should not learn $\gamma^{(n)}_k$, $1 \leq k \leq K$.

Security Model: we model the ISP as an *honest-but-curious* entity, that executes the protocol truthfully but tries to obtain as many information as possible from its transcripts (e.g., the ISP may try to infer the popularity of a content from the secrets' shares that it receives). A variation of the protocol that can deal with a dishonest ISP (i.e., an ISP that lies in its inputs) is described in Section V-F, where we present a subprotocol managed by the RA to perform anti-cheating operations.

B. CONTENT PROVIDERS

We consider K CPs, referred to as CP_k, $1 \leq k \leq K$. A generic CP_k offers a catalogue of contents $\mathcal{C}_k$, which is assumed to be completely stored on a datacenter located outside the area of the ISP. As proposed in [11], each CP remotely manages its portion of cache storage (e.g., by selecting the contents to be cached) and directly serves its users from the cache. Without loss of generality, we assume that the catalogues of the K CPs do not have any content in common (i.e., single catalogues' entries do not overlap). Moreover, we assume that the catalogues of the K CPs are not equally attractive towards the users, i.e., some catalogues are much more popular than others [23] and that users can retrieve contents from any of such catalogues. We refer to the overall catalogue (i.e., the composition of all the CPs catalogues) to as $\mathcal{C}$.

Caching Objectives: a CP aims to maximize its personal Hit-Rate (i.e., the percentage of requests directed to it that are served from the caches), as this results in an improvement of the overall QoE that it can offer to its users [2].

Privacy Requirements: we assume that the CPs aim to protect the following information:

1) *Confidentiality of the requests*: given the generic request r issued by user u toward CP_k, the ISP, the RA and all the CPs (except CP_k itself) should not be able to identify the requested content with non-neglibigle probability.
2) *Contents' popularity*: given two contents c_x and c_y, the ISP, the RA and all the CPs should not be able to say if c_x is more popular than c_y with non-negligible probability. In case both c_x and c_y belong to the generic CP_k, only that CP can know which content is more popular than the other. It is important to remark that disclosing the information about contents' popularity would

reveal extremely confidential insights about the competition between the CPs (e.g., how the market shares are distributed among the CPs).

3) *Number of contents and their size*: the ISP, the RA and all the CPs should not be able to discover the total number of contents owned by the CPs, as well as their sizes.

Security Model: our protocol guarantees a popularity-driven subdivision of the storage, but its effectiveness is based on the assumption that CPs honestly execute it. In fact, if CPs altered their data during the execution of the protocol (e.g., by lying about a requested content), the obtained subdivision would not reflect the correct proportion among CPs' popularity. Driven by the idea that each CP has scarce knowledge about the popularity patterns of the competitors, we assume that it is also not able to alter its data in such a way to obtain a portion of cache storage larger that what it is entitled to receive. Moreover, we assume that the CPs do not have the economical incentives to collude with each other. Hence, CPs can be considered *honest*.

C. REGULATOR AUTHORITY

The Regulator Authority (RA) is considered a *honest* entity that engages with the ISP and the CPs only the legitimate exchange of information envisioned by the protocol. The RA has the main objective of ensuring a NN-compliant storage subdivision (i.e., popularity-driven) division and acts as a guarantor that CPs' and ISP's privacy is not violated. Moreover, the RA is the only entity that knows the private key that can be used to decrypt the data ciphered with the Paillier cryptosystem.

V. THE NN-COMPLIANT PROTOCOL

The NN-compliant protocol involves a set of operations that are mainly performed over the shares of the secrets that ISP, CPs and RA generate using SSS and exchange among each other. We consider a (2,2) SSS, i.e., only a collusion of 2 out of 2 participants allows to reconstruct the secrets.

The protocol works in four main phases: *preliminary operations*, *share collection*, *operations on shares* and *caching*. The preliminary operations are needed to make the parties learn data (e.g., the shares of the multiplication triples) that will be needed during the execution of the protocol. Hence, such operations can be performed in an off-line fashion. The successive three phases last for a period of T_{col}, T_{op} and $T_{caching}$, respectively, and are cyclically repeated as depicted in Figure 2. In the same figure it is also possible to notice that the share collection and the operations on shares phases start simultaneously after the end of the previous share collection phase and that, by construction, $T_{col} = T_{caching}$. We describe the aforementioned phases in the following subsections.

A. PRELIMINARY OPERATIONS

Preliminary operation aim is to give the ISP the information on the average size of CPs' contents and to compute the shares of the multiplication triples required to perform secret multiplications.

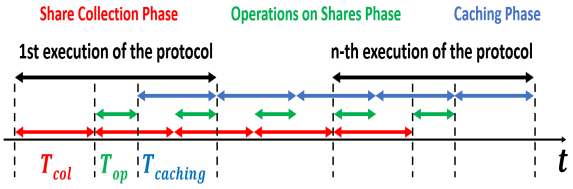


FIGURE 2. Phases of the execution of the NN-compliant protocol.

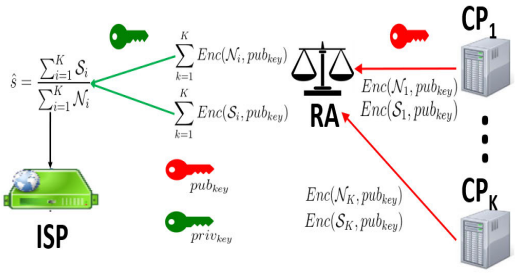


FIGURE 3. Secure computation of the average contents' size.

1) SECURE COMPUTATION OF THE AVERAGE DIMENSION OF THE CONTENTS

First, the ISP learns $\hat{s}$, i.e., the average size of the contents owned by the CPs. This value is needed to obtain the average number of contents that a cache can store (i.e., N_{cache}) from its size S_{cache} (see Eq. 1). This phase is designed to allow the CPs to not disclose to the ISP neither the number nor the size of their contents. The k-th CP uses the public key pub_k to encrypt (i) the sum of the sizes of its contents (i.e., $\mathcal{S}_k$) and (ii) the number of contents of its catalogue (i.e., $\mathcal{N}_k$) by means of the Paillier cryptosystem briefly reviewed in Section II-A. Both $Enc(\mathcal{S}_k)$ and $Enc(\mathcal{N}_k)$ are sent to the RA. This operation is performed by all the K CPs. Then, the RA computes $\sum_{k=1}^{K} Enc(\mathcal{N}_k)$ and $\sum_{k=1}^{K} Enc(\mathcal{S}_k)$ that, due to the additive homomorphic properties of the Paillier cryptosystem, correspond to the encryption of the total number of contents and to the overall summation of their sizes, respectively. The RA, which is assumed to be the only entity who knows the private key $priv_k$, successively decrypts the two values and obtain $\sum_{k=1}^{K} \mathcal{N}_k$ and $\sum_{k=1}^{K} \mathcal{S}_k$. From these values, it is then simple to compute the average size of the contents as $\hat{s} = \frac{\sum_{k=1}^{K} \mathcal{S}_k}{\sum_{k=1}^{K} \mathcal{N}_k}$, which is sent by the RA to the ISP. A representation of this phase is depicted in Fig. 3.

2) SECURE COMPUTATION OF A MULTIPLICATION TRIPLE

The ISP and the RA compute the shares of a multiplicative triple ($[\![a]\!]_{ISP}$, $[\![a]\!]_{RA}$,$[\![b]\!]_{ISP}$, $[\![b]\!]_{RA}$,$[\![c]\!]_{ISP}$, $[\![c]\!]_{RA}$ such that $c = a \cdot b$) by means of the scheme presented in [19] and briefly reviewed in Section II.

B. COLLECTION OF THE SHARES

Upon a new request (say r_i) for content $c_j \in \mathcal{C}$, the owner CP generates two shares of the identifier (e.g., the name) of

content c_j, i.e., $[\![r_i]\!]_{ISP} = [\![c_j]\!]_{ISP}$ and $[\![r_i]\!]_{RA} = [\![c_j]\!]_{RA}$, and sends them to the ISP and the RA, respectively. We assume that the ISP and the RA can always associate a share with the owner CP (e.g., by means of its IP address). At the end of this phase, the ISP and the RA know the shares of all the requests $\mathcal{R}$ issued during the share collection phase, i.e., $\mathcal{S}_{ISP} = \{[\![r_i]\!]_{ISP}\}$ and $\mathcal{S}_{RA} = \{[\![r_i]\!]_{RA}\}$, $\forall r_i \in \mathcal{R}$. Notice that, even if the same content c_j is requested in both r_1 and r_2, it holds that $[\![r_1]\!] \neq [\![r_2]\!]$, which prevents the ISP from inferring the popularity patterns of the CPs. The operations performed in this phase are shown in Subprotocol 1.

Subprotocol 1 Collecting the Shares of the Requested Contents' Identifiers

Input: RA: None
ISP: None
CPs: Each CP_k inputs the subset of contents' requests $\mathcal{R} = \{r_1, \ldots, r_N\}$ directed to it
Output: RA learns $[\![r_i]\!]_{RA}$, $1 \leq i \leq N$
ISP learns $[\![r_i]\!]_{ISP}$, $1 \leq i \leq N$
CPs learn nothing

1: **for** $1 \leq i \leq N$ **do**
2: Let CP_k be the owner of the content requested in r_i
3: CP_k generates $[\![r_i]\!]_{ISP}$ and $[\![r_i]\!]_{RA}$
4: $CP_k \rightarrow$ RA: $[\![r_i]\!]_{RA}$
5: $CP_k \rightarrow$ ISP: $[\![r_i]\!]_{ISP}$
6: **end for**

C. OPERATIONS ON SHARES

Since the ISP and the RA perform the same operations on their set of shares, we omit the apex unless necessary and we describe the operations performed over the abstract set of shares $\mathcal{S} = \{[\![r_i]\!], \forall r_i \in \mathcal{R}\}$ that have been collected during the collection phase. The operations performed over the shares are shown in Subprotocol 2 and described in the following:

1) AGGREGATE IF EQUAL

Given a set $\mathcal{S} = \{[\![r_i]\!], \forall i \in \mathcal{R}\}$ containing the shares relative to the contents of N requests, the objective of this phase is to obtain the share $[\![n_i]\!], \forall i \in \{1, \ldots, N\}$, where n_i is total number of requests of the content requested in r_i. To perform this operation, we employ the algorithm presented in [13] and briefly reviewed in Section II, that computes the aggregation of a set of N elements (in the form of secret shares of key and value) by recursively executing the *aggregate-if-equal* algorithm $N \log N$ times. In our application of the protocol, the key is the share of the content c_j hidden in the i-th request r_i, i.e., $[\![r_i]\!] = [\![c_j]\!]$, while the value associated is the share of 1 for all the requests. Since both the ISP and the CPs might be interested in altering the value (as this would favour some contents over others and ultimately affect the caching process), we mandate the RA to generate $[\![1]\!]_{RA}$ and $[\![1]\!]_{ISP}$ at each request. At the end of this phase, the ISP and the RA obtain the respective shares of $[\![n_i]\!], \forall i \in \{1, \ldots, N\}$.

Subprotocol 2 Performing Operations on the Shares

Input: RA: $[\![r_i]\!]_{RA}$, $1 \leq i \leq N$
ISP: $[\![r_i]\!]_{ISP}$, $1 \leq i \leq N$
Output: RA learns $[\![z_k]\!]_{RA}$, $1 \leq k \leq K$
ISP learns $[\![z_k]\!]_{ISP}$, $1 \leq k \leq K$

1: RA generates two shares of the constant 1: $[\![1]\!]_{RA}$ and $[\![1]\!]_{ISP}$
2: ISP generates two shares of the constant N_{cache}: $[\![N_{cache}]\!]_{RA}$ and $[\![N_{cache}]\!]_{ISP}$
3: RA $\rightarrow$ ISP: $[\![1]\!]_{ISP}$
4: ISP $\rightarrow$ RA: $[\![N_{cache}]\!]_{RA}$

RA and ISP locally execute for j = RA and j = ISP, respectively

1: Execute the *aggregate-if-equal* algorithm over the set $\{([\![r_i]\!]_j, [\![1]\!]_j), 1 \leq i \leq N\}$ and obtain $\{[\![n_i]\!]_j, 1 \leq i \leq N\}$
2: Execute the *comparison* algorithm between all the pairs of elements of the set $\{[\![n_i]\!]_j, 1 \leq i \leq N\}$ and obtain $\{[\![\pi_i]\!]_j, 1 \leq i \leq N\}$
3: **for** $1 \leq i \leq N$ **do**
4: Execute the *comparison* algorithm on $([\![\pi_i]\!]_j, [\![N_{cache}]\!]_j)$ and obtain $[\![\beta_i]\!]_j$
5: The CP_k to which r_i is directed is identified
6: Updating of the number of contents belonging to CP_k whose popularity rank $\pi < N_{cache}$: $[\![z_k]\!]_j \leftarrow [\![z_k]\!]_j + [\![\beta_i]\!]_j$
7: **end for**

2) RANK COMPUTATION

From the previous phase, ISP and RA have obtained a set $[\![n_i]\!], \forall i \in \{1, \ldots, N\}$ containing the shares of the number of occurrences for each requested content. With these data in hand, they aim at computing $\pi_i \in [0, N-1]$, i.e., the rank of the content requested in r_i, where π_i increases with decreasing popularity of the associated content (i.e., $\pi = 0$ for the most popular content).

To perform this task, all the shares $[\![n_i]\!], \forall i \in \{1, \ldots, N\}$ need to be compared with each other, for a total of N^2 executions of the *comparison* algorithm mentioned in Section II. We recall that the algorithm takes in input two shares $[\![x_1]\!]$ and $[\![x_2]\!]$ and returns $[\![1]\!]$ if $x_1 \leq x_2$, and $[\![0]\!]$ otherwise. Considering that $[\![1]\!] = 1 - [\![0]\!]$ and $[\![0]\!] = 1 - [\![1]\!]$ (due to the additive homomorphic properties of SSS), it is possible to assign the share $[\![1]\!]$ to the lower value (say x_1) and the share $[\![0]\!]$ to the higher one (say x_2) with a single execution of the comparison algorithm. Hence, the complexity is reduced from N^2 to $\binom{N}{2}$ executions of the *comparison* algorithm. The rank π_i can then be computed by summing up the results, in secret shared form, of the relative comparisons as $[\![\pi_i]\!] = \sum_{x=1, x\neq i}^{N-1} [\![l_{i,x}]\!]$, where $l_{i,x} = 0$ if $n_i \leq n_x$ (and 1 otherwise). Notice that, if r_i is the request relative to the most popular content, then $l_{i,x} = 0, \forall x$ (because its number of occurrences is higher than all the others) and, as expected, $\pi_i = 0$.

Once the shares of ranks $[\![\pi_i]\!], \forall i \in \{1, \ldots, N\}$ have been obtained, ISP and RA need to compute the share of the portion

of cache that each CP is expected to receive. To this aim, the rank of each content needs to be compared with the size of the cache and, if $\pi_i \leq N_{cache}$, the CP to which r_i is directed is entitled to store one content. Since the ISP wants to protect the information about its cache size, it generates 2 shares $[\![N_{cache}]\!]_{ISP}$ and $[\![N_{cache}]\!]_{RA}$, which can be used to perform a comparison with $[\![\pi_i]\!], \forall i \in [1, \ldots, N]$ by means of the *comparison* algorithm. The result of the i-th comparison is $[\![\beta_i]\!]$, with $\beta_i = 1$ if $\pi_i \leq N_{cache}$ (and 0 otherwise).

By repeating this operation for all the requests directed towards CP_k, i.e., $\mathcal{R}_k$, ISP and RA obtain the share of the number of most popular contents owned by CP_k as $[\![z_k]\!] = \sum_{i \in \mathcal{R}_k} [\![\beta_i]\!]$.

D. CACHING

Subprotocol 3 Calculating the Portion of Cache Storage to Allocate to Each CP

Input: RA: $[\![S_{cache}]\!]_{RA}, [\![z_k]\!]_{RA}, 1 \leq k \leq K$
ISP: $[\![S_{cache}]\!]_{ISP}, [\![z_k]\!]_{ISP}, 1 \leq k \leq K$
CPs: None
Output: RA learns nothing
ISP learns $\gamma_k = \frac{rv \cdot z_k \cdot S_{cache}}{rv \cdot \sum_{j=1}^{K} z_j}, 1 \leq k \leq K$
Each CP_k learns $\gamma_k = \frac{rv \cdot z_k \cdot S_{cache}}{rv \cdot \sum_{j=1}^{K} z_j}$

1: ISP and RA compute the shares $[\![rv]\!]_{ISP}$ and $[\![rv]\!]_{RA}$ of a secret random integer rv using the scheme of [14]
2: ISP and RA compute $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{ISP}$ and $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{RA}$ using the multiplication protocol of [18]
3: **for** $k \in \{1, \ldots, K\}$ **do**
4: RA $\rightarrow CP_k$: $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{RA}$
5: ISP computes $[\![rv \cdot z_k \cdot S_{cache}]\!]_{ISP}$
6: RA computes $[\![rv \cdot z_k \cdot S_{cache}]\!]_{RA}$
7: RA $\rightarrow CP_k$: $[\![rv \cdot z_k \cdot S_{cache}]\!]_{RA}$
8: **end for**
9: **for** $k \in \{1, \ldots, K\}$ **do**
10: $CP_k \rightarrow$ ISP: $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{RA}$ and $[\![rv \cdot z_k \cdot S_{cache}]\!]_{RA}$
11: ISP $\rightarrow CP_k$: $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{ISP}$ and $[\![rv \cdot z_k \cdot S_{cache}]\!]_{ISP}$
12: **end for**
13: **for** $k \in \{1, \ldots, K\}$ **do**
14: ISP and CP_k reconstruct the secret $rv \cdot \sum_{j=1}^{K} z_j \leftarrow \left([\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{RA}, [\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{ISP}\right)$
15: ISP and CP_k reconstruct the secret $rv \cdot z_k \cdot S_{cache} \leftarrow \left([\![rv \cdot z_k \cdot S_{cache}]\!]_{RA}, [\![rv \cdot z_k \cdot S_{cache}]\!]_{ISP}\right)$
16: **end for**
17: **for** $k \in \{1, \ldots, K\}$ **do**
18: ISP computes $\gamma_k = \frac{rv \cdot z_k \cdot S_{cache}}{rv \cdot \sum_{j=1}^{K} z_j}$
19: CP_k computes $\gamma_k = \frac{rv \cdot z_k \cdot S_{cache}}{rv \cdot \sum_{j=1}^{K} z_j}$
20: **end for**

We remind that CP_k is entitled to receive a portion of storage $\gamma_k = \frac{z_k}{\sum_{j=1}^{K} z_j} \cdot S_{cache}$. The ISP and the RA know their shares $[\![S_{cache}]\!]$ and $[\![z_k]\!], 1 \leq k \leq K$ and could recover z_k, $\sum_{j=1}^{K} z_j$ and S_{cache} and obtain from them the value γ_k. The exchange of shares and the operations performed on them to compute the cache storage subdivision are described in the following and shown in Subprotocol 3.

However, we prevent the ISP and the RA from directly reconstructing these secrets since (i) from $\sum_{j=1}^{K} z_j$ the RA could obtain a good estimate of the size of the cache S_{cache} and (ii) from $z_k, 1 \leq k \leq K$ it would discover the number of contents owned by each CP whose popularity rank is less than N_c. Instead, ISP and RA employ the scheme described in [14] to obtain the shares of a random integer $[\![rv]\!]_{ISP}$ and $[\![rv]\!]_{RA}$ without learning rv itself. With these values in hands, they then learn, using the multiplicative protocol proposed in [18], $[\![rv \cdot z_k \cdot S_{cache}]\!], 1 \leq k \leq K$ and $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]$. Notice that these values represent the shares of the numerator and denominator of $\frac{z_k}{\sum_{j=1}^{K} z_j}$, respectively, which have been masked with the same value rv to keep the ratio between them unchanged (and equal to γ_k).

Then, the RA sends $[\![rv \cdot z_k \cdot S_{cache}]\!]_{RA}$ and $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{RA}$ to the corresponding k-th CP, which exchanges with the ISP their shares to recover $rv \cdot z_k \cdot S_{cache}$ and $rv \cdot \sum_{j=1}^{K} z_j$. From these two reconstructed secrets, both the ISP and the k-th CP compute the amount of storage destined to CP_k:

$$\gamma_k = \frac{rv \cdot z_k \cdot S_{cache}}{rv \cdot \sum_{j=1}^{K} z_j} \tag{2}$$

Notice that the k-th CP learns nothing more that its allocated storage. For example, it does not learn the percentage of storage it is assigned, from which it would have derived the size of the cache S_{cache}.

At this point, the k-th CP can start caching its contents in the received storage portion. Notice also that, whilst the popularity-based caches' subdivision is computed by performing operations on the shares relative to contents' requests (i.e., the proposed protocol is designed to work at the content level) caching strategies are successively applied by the CPs on a chunk-level basis, as further described in Section VI-A. In Fig. 4 we depict the most salient shares that the involved parties exchange with each other during the last three phases of execution of the protocol, namely collection of shares, operations on shares and caching.

E. FULLFILMENT OF PRIVACY REQUIREMENTS

1) ISP'S PRIVACY REQUIREMENTS

We remind that neither the RA nor the CPs are allowed to obtain the size of the ISP's caches (i.e., S_{cache}) and that the RA is not allowed to obtain the portion of storage given to CP_k, i.e., $\gamma_k, 1 \leq k \leq K$.

During the execution of the protocol (in the rank computation phase, precisely) the RA obtains the share $[\![N_{cache}]\!]$. Since SSS is proved secure under the information-theoretic security model [17], this share provides absolutely no additional information on the relative secret. Hence, the RA does not discover the size of the cache. Then, in the caching phase, the RA learns $[\![rv \cdot z_k \cdot S_{cache}]\!]$ and $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]$. Under the

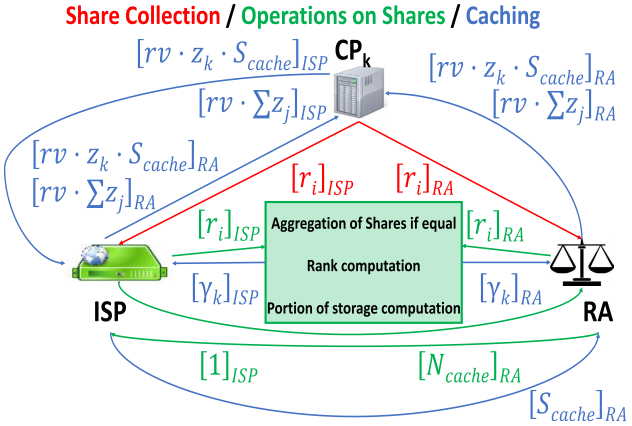


FIGURE 4. Main shares learnt by ISP, CPs and RA during the execution of the NN-compliant protocol.

assumption of honest RA, ISP and RA do not exchange their shares with each other. Hence, the RA does not obtain γ_k.

During the caching phase, $CP_k, , 1 \leq k \leq K$ learns $rv \cdot z_k \cdot S_{cache}$ and $rv \cdot \sum_{j=1}^{K} z_j$, from which it computes $\gamma_k = \frac{rv \cdot z_k \cdot S_{cache}}{rv \cdot \sum_{j=1}^{K} z_j}$. Notice that the ability of CP_k to estimate S_{cache} is bounded by its ability to assess its popularity with respect to the popularity of its competitors, which is encoded in the ratio $\frac{z_k}{\sum_{j=1}^{K} z_j}$. Since we have assumed that each CP has scarse knowledge about other CPs' attractiveness, we consider S_{cache} to be protected from the CPs as well.

2) CP'S PRIVACY REQUIREMENTS

Considering the first privacy requirement of the CP, i.e., the confidentiality of the requests, in the share collection phase the ISP and the RA receive $[\![r_i]\!]_{ISP}$ and $[\![r_i]\!]_{RA}$ from the CP towards which the i-th request is issued (say CP_k). As SSS is information-theoretically secure, it holds that:

$$P(c_j|[\![r_i]\!]) = \frac{1}{\mathcal{N}_k}, \quad \forall j \in \{1, \ldots, \mathcal{N}_k\} \tag{3}$$

where $P(c_j|[\![r_i]\!])$ is the probability that the share $[\![r_i]\!]$ refers to content c_j and $\mathcal{N}_k$ is the total number of contents owned by CP_k. No CP (except CP_k) obtains $[\![r_i]\!]$ from the execution of the protocol and both the RA and the ISP can identify the content hidden behind the i-th request with a negligible probability only. Hence, the first CPs' privacy requirement is fulfilled.

Concerning the second privacy requirement, i.e., protection of contents' popularity, in the rank computation phase the ISP and the RA obtain the shares of the number of occurrences of the content hidden behind the i-th request, i.e., $[\![n_i]\!], \quad \forall i$. They then compare the number of occurrences of each pair of contents (say c_i and c_x) and obtain the result in secret shared form (i.e., $[\![l_{i,x}]\!]$). Due to the information-theoretically security properties of SSS, $P(l_{i,x} = 0) = P(l_{i,x} = 1) = 0.5$. Hence, neither the ISP nor the RA can violate the privacy of contents' popularity.

Finally, to satisfy the third privacy requirement, the ISP should not obtain the number and the sizes' of CPs' contents. During the preliminary operations, the ISP only obtains the average size of contents $\hat{s}$, from which it cannot derive neither the total number of contents, nor their sizes.

F. EXTENSION OF THE PROTOCOL FOR DISHONEST ISP

In this Section, we describe a scenario in which, by maliciously forging its data, the ISP can obtain an unfair subdivision of the cache storage. We then provide an extension of the protocol to make the RA able to discover if the ISP is cheating.

The generic cache server is characterized by its size S_{cache} and by the average number of contents that it can store N_{cache}, according to the relation $S_{cache} = N_{cache} \cdot \hat{s}$, being $\hat{s}$ the average size of CPs' contents. N_c determines the number of contents that the CPs regard as the most popular ones. Just as an example, let us think of the case of 2 CPs, referred to as CP_1 and CP_2, which own contents whose popularity ranks go from 0 to 49 and from 50 to 99, respectively. If $N_{cache} = 50$ contents, then, according to our definition of NN-compliant caching, the 100% of the total cache storage should be assigned to CP_1. This value drops to 50% if, instead, $N_{cache} = 100$ contents. This scenario shows that, by communicating to the RA the share of a forged N_{cache}, the ISP is able favour a specific CP. To address this issue, the RA can compare $\hat{s} \cdot [\![N_c]\!]_{RA}$ and $[\![S_c]\!]_{RA}$ using the equality-test operator, and ask the ISP to perform a similar operation. By doing so, RA and ISP learn the shares $[\![b_{eq}]\!]_{RA}$ and $[\![b_{eq}]\!]_{ISP}$, from which they recover b_{eq} that is equal to 1 if the ISP did not forged N_{cache}, and 0 otherwise.

VI. DYNAMIC SIMULATIONS FOR VOD CONTENT CACHING AND DISTRIBUTION

To evaluate the performance of the proposed privacy-preserving network-neutrality compliant caching protocol, we develop a discrete-event-driven simulator to perform dynamic simulations of VoD content caching and distribution. In this section, we describe the developed simulator, the VoD request provisioning process and the general simulation settings.

A. DYNAMIC VoD CONTENT CACHING AND DISTRIBUTION SIMULATOR

The overall framework of the simulator is described as follows: Given the network topology, content catalogue characteristics of each CP, locations of caches and the list of stored contents per CP in each cache, the simulator provisions the dynamically-arriving VoD-content requests, based on current network status, and gives as an output the overall amount of resources occupied to provision contents of a specific CP, the overall RO of the network and caches' hit-ratios.

Note that a VoD-content request is provisioned taking into consideration its chunk-nature, i.e., each VoD request, according to its duration, consists of a number of chunks and the chunks are provisioned sequentially. This allows to

TABLE 1. Table of notations.

Protocol's variables and relative description		Simulation settings's Variables and relative description	
Variable	**Description**	**Variable**	**Description**
r_i	i-th request issued from users	N	Total number of contents' requests
$\mathcal{R} = \{r_i\}$	Set of all the requests	M	Total number of available contents
$[\![x]\!]$	Share of the generic value x	$\mathcal{N}_k$	Number of contents of CP_k's catalogue
n_i	Number of requests for the content hidden behind the i-th request	$\mathcal{S}_k$	Sum of the sizes (measured in bytes) of the CP_k's contents
π_i	Popularity rank of the content hidden behind the i-th request	$\hat{s}$	Average size (measured in bytes) of the M available contents
z_k	Number of contents belonging to CP_k whose popularity rank is below N_{cache}	$N_{cache}^{(n)}$	Capacity of the generic n-th ISP's cache (measured as the average number of contents that can be stored on it)
γ_k	Amount of cache storage allocated to CP_k (measured in bytes)	$S_{cache}^{(n)}$	Capacity of the generic n-th ISP's cache (measured in bytes)
$a, b, c \vert c = a \cdot b$	Multiplicative triple securely shared during the preliminary operations	α	Skewness parameter of the contents' popularity distribution
$pub_{key}, priv_{key}$	Public and private keys used to securely compute the average contents' size	ϕ	Bit-length representation of a share

have different chunks of the same VoD request delivered from different caches, which is basically the case when caches are dynamically updated, i.e., when contents are pulled out from or pushed in caches. Specifically, a VoD-chunk request is described by the tuple r = (t_r, D_r, b_r, m, d_r), where t_r is the request arriving time from node D_r, b_r is the requested bit-rate, m is the requested content and d_r is the chunk duration. The simulated VoD-chunk provisioning/deprovisioning process is described as follows: Upon arrival of a VoD-chunk request for content m from node D_r, a list of all cache nodes hosting m (including the video server) is identified. Then, the nearest cache storing content m delivers the chunk to node D_r, considering a path with available bandwidth greater than or equal to b_r. The chunk is later deprovisioned at time $t_s + d_r$ deallocating the assigned bandwidth from the utilized path.

B. NETWORK MODEL AND CACHING SYSTEM

We consider a real ISP metro-aggregation network topology, depicted in Fig. 5. The network consists of three types of nodes, namely metro-core backbone nodes, metro-core nodes and metro-aggregation nodes. We assume that the metro-core and metro-aggregation nodes are cache-enabled nodes, i.e., capable of hosting and delivering video contents while the metro-core backbone nodes are routers connecting the ISP to the Internet. As for the cache-enabled nodes, we considered 2 metro-core and 12 metro-aggregation caches whose locations are highlighted in Fig. 5.

C. TRAFFIC MODEL

Information about contents' requests is widely considered sensitive and business relevant by CPs. Hence, public data sets are rarely available to the research community and we had to perform our simulations over synthetic traffic traces, which have been crafted as follows. Based on a common assumption made in the literature, we consider a fixed catalogue [24] of contents whose popularities are distributed according to the Zipf law, i.e., $p_j = \frac{j^{-\alpha}}{\sum_{z=0}^{M-1} p_z}, \forall j \in$

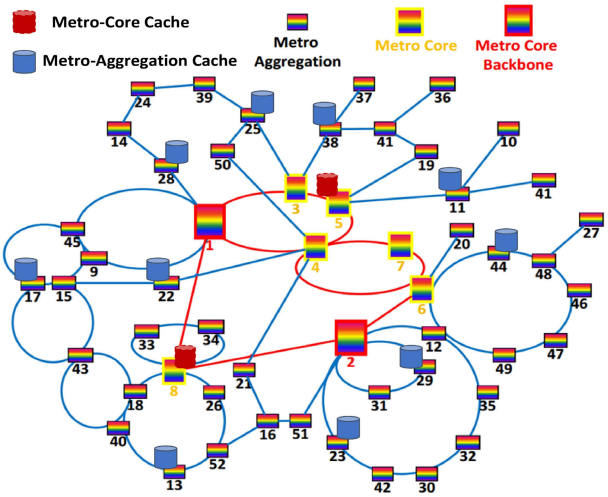


FIGURE 5. Schematic representation of the ISP Network Topology and the location of caches.

$\{0, \ldots, M-1\}$, where p_j is the probability that the j-th popular content is chosen among the available M videos. $\alpha \in [0, 1]$ is the skew parameter of the Zipf (the number of scarcely-requested contents increases with increasing α). Inspired by [25], we also introduce a temporal dynamic to this popularity distribution. In particular, every 30 minutes we sum (or subtract, with the same probability) a Poisson-distributed random variable (with mean value 1) to the popularity rank of each content $c_j, \forall j \in \{0, \ldots, M-1\}$. Notice that the described catalogue results from the aggregation of the single catalogues owned by each CP.

Finally, we consider CPs that offer, on average, contents of significantly different popularities. Although being a well-known characteristic of existing CPs (few of which are much more popular than the others), to our knowledge a contents' popularity model that take this fact into consideration has never been proposed in the literature. To fill this gap, we propose a model that is descibed in the following. We assume

that the k-th CP is characterized by a gaussian probability distribution $\rho_j^{(k)}$ over the ranks of the overall catalogue with mean value $\mu_k = \frac{M}{K} \cdot (k - \frac{1}{2})$ and standard deviation $\sigma_k = \sigma_1 + (K - k) \cdot \frac{\sigma_2 - \sigma_1}{K}$, where M and K are the total number of contents and of CPs, respectively. σ_1 and σ_2 are tuned to obtain different degrees of CPs' popularity, in particular to model the difference of CPs' attractiveness towards the users. To this aim, in Section VII we consider scenarios with different values of σ_1 and σ_2.

According to the proposed model, the j-th popular content of the overall catalogue described above belongs to the k-th CP with probability $P(j,k) = \frac{\rho_j^{(k)}}{\sum_{x=1}^{K} \rho_j^{(x)}}$. In this way, for example, given $K = 5$ CPs and $M = 25000$ contents and considering $\sigma_1 = \frac{M}{K}$ and $\sigma_2 = \frac{\sigma_1}{K}$, CP_1 and CP_5 are assigned contents with an average rank of 4369 and 22144, and standard deviations of 3348 and 1946, respectively. This makes the contents offered by CP_5 much less popular than those owned by CP_1, on average.

We assume that the duration of the contents is a random variable distributed according to a Pareto distribution with skew parameter equal to 0.25. All the durations are then normalized between 1200 s and 8400 s. We then assume the same bit-representation for all the contents to be equal to 12 Mbits (hence contents have a size that ranges between 1.8 and 12.6 Gbytes).

VII. ILLUSTRATIVE SIMULATIVE RESULTS

A. SIMULATION SETTINGS

In our experiments, we consider three approaches of cache storage subdivision, namely the popularity-driven, the resource-occupation-driven and the static subdivisions. In the first approach, which is enabled by the use of our protocol, each CP receives a portion of storage proportional to the popularity of its contents. In the second approach each CP receives a portion of storage proportional to the RO the the delivery of its contents generates within the network of the ISP. In the third approach, all the CPs receive the same amount of storage.

To compare the performance of these approaches, we perform simulations on two different scenarios, the first characterized by $K = 5$ CPs and $\hat{M} = 5000$, and the second by $K = 10$ CPs and $\hat{M} = 5000$, for values of $T_{col} \in \{10, 20, 30, \ldots, 100\}$ minutes. In each simulation, we simulate the arrival of 43000 VoD requests generated according to the traffic model described in Sec. VI-C at an arrival rate guaranteeing negligible blocking probability (i.e., Zipf $\alpha = 0.8$ and $\lambda = 1\ req/sec$), to provide a fair comparative analysis between the considered approaches. We assume the network topology shown in Fig. 5 with cache locations highlighted. We fix the size of caches located at metro-aggregation nodes and those located at metro-core nodes to 5% and 10% of the overall content catalogues size (of all content catalogues of all CPs).

B. DISCUSSION

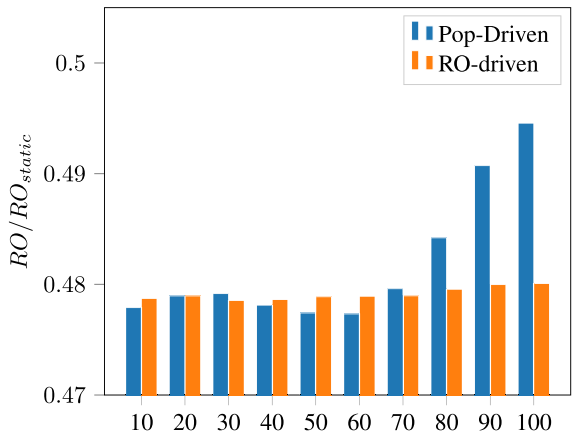
1) ISP'S RESOURCE OCCUPATION AND CACHING HIT-RATE

In this section, we show the comparison of popularity-driven, resource-occupation-driven and static subdivisions considering the overall network RO and the Hit-Rate measured by the ISP for increasing T_{col}.

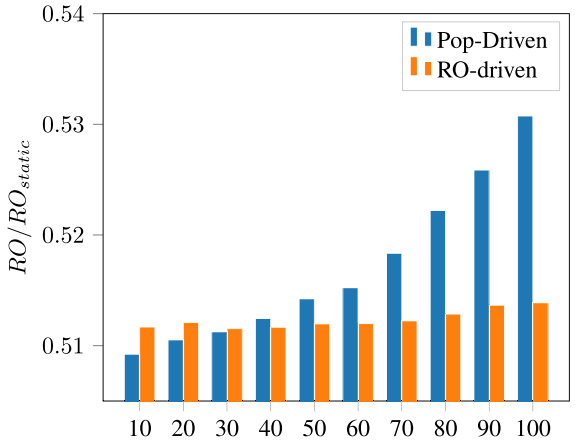
First, we depict the RO obtained with the former approaches as a percentage of the RO measured when the static subdivision is enforced (which is equal to $\sim 760 \cdot 10^6 Mbit$ if $K = 5$ CPs and $\sim 713 \cdot 10^6 Mbit$ if $K = 10$ CPs). The $\frac{RO}{RO_{static}}$ as a function of T_{col} is depicted in Fig. 6(a) and Fig. 6(b), for the scenarios with 5 and 10 CPs, respectively. We remind that an approach is preferable to the ISP if it reduces the RO measured within its network. We note that both the popularity-driven and the resource-occupation-driven subdivision lead to a remarkable RO gain with respect to the static subdivision. In both the scenarios under analysis, the minimum RO is obtained when the storage of the caches is divided according to the popularity-driven subdivision. More specifically, the minimum RO is obtained with $T_{col} = 10$ minutes and at $T_{col} = 50$ minutes when 5 CPs and 10 CPs are considered, respectively. This result confirms that the effectiveness of caching highly depends on information about contents' popularity and motivates the adoption of our protocol as a tool to keep this information private.

In general, we observe a RO increase for increasing T_{col}. This fact can be explained considering that high values of T_{col} allow the ISP to obtain more information (e.g., about contents' popularity), but, at the same time, increases the number of changes that contents' popularity undergo during T_{col}. This increase is much more evident in the popularity-driven subdivision, with such percentage going from $\sim 50.9\%$ to $\sim 53\%$ when T_{col} passes from 10 to 100 minutes, whereas the percentage increases only slightly and it is mostly stable around $\sim 51\%$ when the resource-occupation-driven subdivision is employed. This difference between the two approaches is due to fact that our protocol introduces a delay between the computation of the popularity-driven subdivision and its actual enforcement. This delay increases with increasing T_{col} and this may make the computed storage subdivision out-of-date with respect to the current popularity patterns (we elaborate further on the dependency between this delay and T_{col} in Section VII-B.3). The conflicting effects of increasing T_{col} are more visible in Fig. 6(a), where it is possible to observe that the RO of the popularity-driven subdivision decreases until the minimum value is reached (at $T_{col} = 50$ minutes) and then increases up to the maximum (at $T_{col} = 100$ minutes).

Fig. 7 shows the Hit-Rates measured at the caches located in the metro-aggregation level (i.e., the percentage of requests served from the caches closer to the users). Obtained results are consistent with the RO previously described: (i) the Hit-Rates of popularity-driven and resource-occupation-driven subdivisions significantly outperform the static subdivision and (ii) the RO decreases

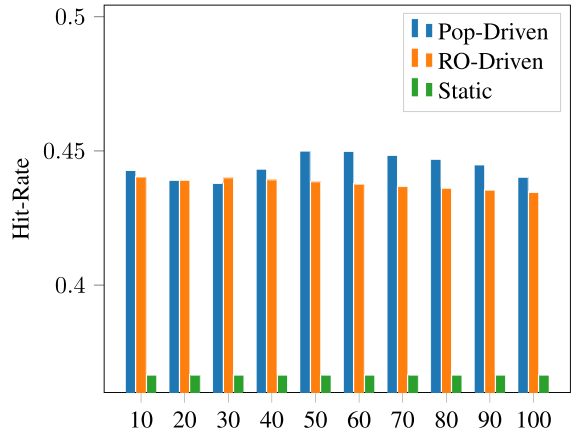


(a) *Comparison of the RO considering a scenario with* 5 *CPs with an average number of* 5000 *contents*

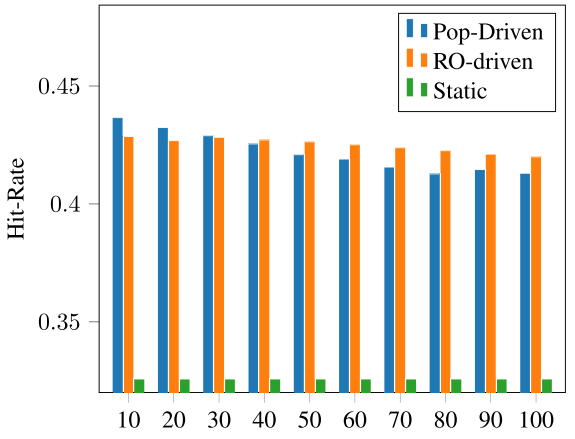


(b) *Comparison of the RO considering a scenario with* 10 *CPs with an average number of* 5000 *contents*

FIGURE 6. Resource Occupation vs T_{col} obtained with the popularity-driven and with the resource-occupation-driven subdivisions divided by the RO achieved with the static subdivision.



(a) *Comparison of the ISP's Hit-Rate considering a scenario with* 5 *CPs with an average number of contents of* 5000



(b) *Comparison of the Hit-Rate considering a scenario with* 10 *CPs with an average number of contents of* 5000

FIGURE 7. ISP's Hit-Rate (measured at metro-aggregation caches) vs T_{col} obtained with the popularity-driven, the resource-occupation-driven and the static subdivisions.

(resp., increases) when the Hit-Rate increases (resp., decreases). The maximum Hit-Rates obtained with a popularity-driven subdivision are higher than the benchmarks in both scenarios. For example, in the scenario with 5 CPs, the maximum Hit-Rates for the popularity-driven and for the resource-driven subdivision are ~ 0.45 and ~ 0.44, respectively (see Fig. 7(a)). When instead 10 CPs are considered, the corresponding values are ~ 0.436 and ~ 0.428 (see Fig. 7(b)).

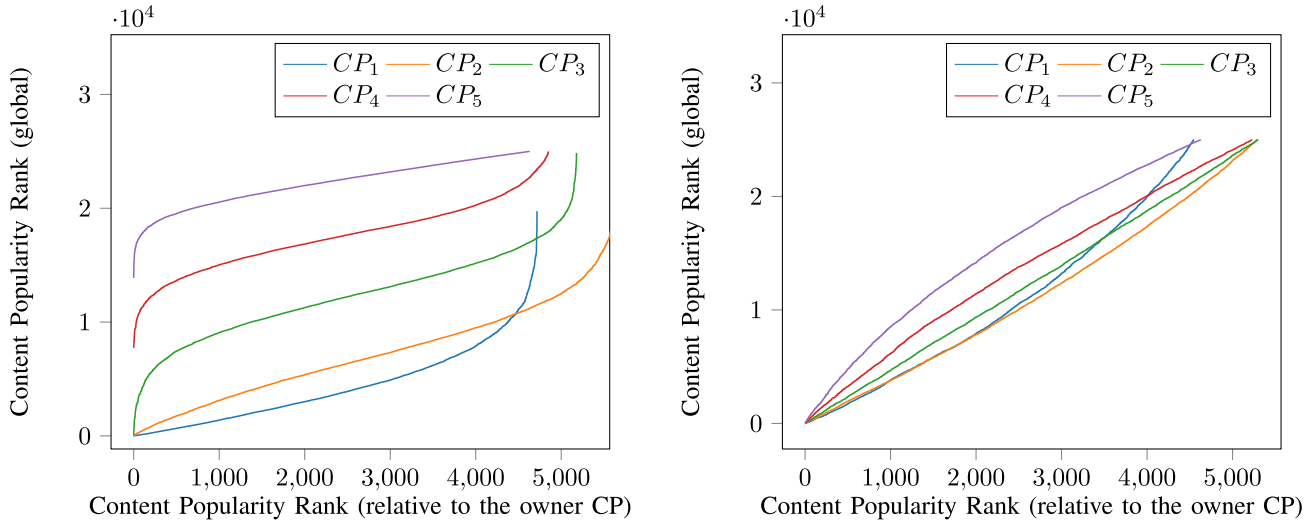
2) HIT-RATES FOR THE CPs

According to our vision of NN an ISP should maximally benefit from the application of caching strategies, as long as they are not discriminatory towards the CPs. Therefore, we believe that the ISP is entitled to decide how frequenty the protocol is executed (i.e., by setting T_{col} to the value that minimizes the RO). However, since the value that minimizes the RO is not necessarily the one that maximizes the hit-ratio of every CP, CPs may experience a loss in their hit-ratio. We formally define this loss as:

$$\mathcal{L}_k = \frac{\hat{h}^{(k)} - \hat{h}^{(k)}_{isp}}{\hat{h}^{(k)}}, \quad 1 \leq k \leq K \tag{4}$$

where $\hat{h}^{(k)}$ is the maximum Hit-Rate that the k-th CP would obtain if it selfishly selected T_{col}, while $\hat{h}^{(k)}_{isp}$ is the Hit-Rate that it actually experiences according to the decision taken by the ISP. Notice that such hit-rates refer to the cumulative hit-rates of metro-aggregation and metro-core caches (i.e., it is the overall percentage of requests that the CPs serve from the area of the ISP).

In Tab. 2, we show the loss for each CPs of the first scenario described in the previous Section (5 CPs with an average



(a) 5 *CPs with an average number of contents of* 5000 *and significantly different attractiveness towards the users*

(b) 5 *CPs with an average number of contents of* 5000 *and similar attractiveness towards the users*

FIGURE 8. **Global popularity of the contents VS Popularity of the contents within the catalogue of the owner CP.**

TABLE 2. **Loss of CPs' Hit-Rates when CPs offer contents with significantly-different popularity.**

CPs	H_{isp}	H_{cps}	Loss
CP_1	0.807	0.811	0.499%
CP_2	0.537	0.556	3.252%
CP_3	0.147	0.171	14.104%
CP_4	0.081	0.084	4.09%
CP_5	0.068	0.094	27.64%

TABLE 3. **Loss of CPs' Hit-Rates when CPs offer contents with similar popularity.**

CPs	H_{isp}	H_{cps}	Loss
CP_1	0.659	0.67	1.65%
CP_2	0.615	0.617	0.31%
CP_3	0.596	0.596	0.03%
CP_4	0.597	0.597	0%
CP_5	0.533	0.533	0%

number of contents of 5000 and contents' popularity distributed as shown in Fig. 8(a)). We notice that the loss highly varies among the CPs that, in this scenario, offer contents of significantly different popularities (e.g., CP_1's contents are much more popular, on average, than CP_5's contents). For instance, the loss goes from a minimum of $\sim$ 0.5% to a maximum of $\sim$ 27.6%, which are experienced by CP_1 and CP_5 (the CP with the most and the least catalogues on average, respectively). In the considered scenario, there is a clear difference of popularity among the CPs. To understand the impact that popularity difference has on the loss, we perform additional simulations on a second scenario in which the contents' popularity is much more similar among the CPs. Also in this second scenario there are 5 CPs offering, on average, 5000 contents. The distribution of contents' popularity is derived setting $\sigma_1 = \sigma_2 = 15000$ and it is depicted in Fig. 8(b). The loss of each CP in this second scenario is presented in Tab. 3, from which we can observe that the loss goes from a minimum of 0% to a maximum of 1.65% and it is therefore much less significant than in the previous case. From this comparison, it becomes evident that the difference in CPs' popularity highly affects the loss experienced by the CPs. This can be explained considering that the hit-ratios of the CPs do not significantly vary with changing the storage subdivision (as a result of tuning T_{col}) if the CPs cache contents with similar popularity. Hence, the hit-ratios of the single CPs do not strongly depend on T_{col} (i.e., the hit-ratios are similar and close to the optimum one regardless the T_{col} chosen by the ISP). We therefore conclude that the CPs are strongly penalized by being inhibited to select T_{col} only if their attractiveness towards the users is significantly different.

3) COMPLEXITY OF THE PROTOCOL AND VOLUME OF THE EXCHANGED DATA

We now provide an evaluation of the data overhead introduced in all the phases of the execution of the protocol, as well as the time needed to perform them.

The secure computation of the average size of CPs' contents' $\hat{s}$ is the only operation executed with our protocol that does not require the use of SSS. The k-th CP $1 \leq k \leq K$ sends to the RA the values of its number of contents $\mathcal{N}_k$ and the overall size of its catalogue $\mathcal{S}_k$ encrypted using the Paillier cryptosystem. Then, the RA decrypts these values and communicates to the ISP the ratio between them, i.e., $\hat{s} = \frac{\sum_{k=1}^{K} \mathcal{S}_k}{\sum_{k=1}^{K} \mathcal{N}_k}$. This operation requires the exchange of $2K$ messages between the CPs and the RA, and the exchange of one piece of data between the RA and the ISP. As all such data have negligible size (i.e., in the order of the hundreds of bits) and their computation is not time-consuming,

TABLE 4. Overhead of data exchanged during the execution of the protocol (being ϕ the bit-length representation of the shares exchanged among the parties).

	ISP	RA	CPs
ISP	0	$\left(\frac{N!}{(N-2)!2!}+N\right)\cdot 9\phi^2$ $+N\log N\cdot 7\phi^2$ $+14\phi$	$4K\phi$
RA	$\left(\frac{N!}{(N-2)!2!}+N\right)\cdot 9\phi^2$ $+N\log N\cdot 7\phi^2$ $+N\phi+14\phi$	0	$2K\phi$
CPs	$N\phi+3K\phi$	$N\phi$	0

the introduced time and data overhead can be considered negligible.

Let us now consider all the remaining operations, which are based on SSS. We refer to $\phi = \lfloor(\log_2 q + 1)\rfloor$ to indicate the bit-length representation of a share $\in \mathbb{Z}_q$. The secure computation of the multiplication triple ($[\![a]\!]$, $[\![b]\!]$, $[\![c]\!]$ such that $c = a \cdot b$) requires the ISP and the RA to exchange 4ϕ bit (to obtain $[\![a]\!]$, $[\![b]\!]$ in a distributed manner) and other 6ϕ to obtain $[\![c]\!]$. The reader is referred to [19] for an in-depth understanding of all the required exchanges.

The collections of the shares relative to N requests issued during T_{col} requires the following exchange of data: $2N\phi$ (to account for the shares sent by the CPs to the ISP and the RA) and $N\phi$ for the shares of 1s sent from the RA to the ISP (to account for the value associated with each request). The next phase requires $N \log N$ equality tests to perform the aggregation of the collected shares and $\binom{N}{2} + N$ comparison operations to compute the ranks of the contents and to compare them with the size of the cache. Each equality operation requires the exchange of $2\phi^2$ (which need to be exchanged during the execution of the protocol, i.e., *online*) and $12\phi^2$ (which can be pre-computed and transmitted before the execution of the protocol, i.e., *offline*) bits, while the comparison operation requires $18\phi^2$ bits exchanged on-line [14].

Then, the ISP and the RA compute the random value rv in a secure and distributed fashion and use it to obtain $[\![rv \cdot z_k \cdot S_c]\!]$ and $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]$. This operations require the exchange of 18ϕ (2ϕ for the computation of rv and 16ϕ needed for the multiplications). Successively, the RA sends the obtained shares $[\![rv \cdot z_k \cdot S_c]\!]_{RA}$ and $[\![rv \cdot \sum_{j=1}^{K} z_j]\!]_{RA}$ to the K CPs, which requires $2K\phi$ additional transmitted bit. Finally, the ISP exchanges with the CPs their shares to obtain γ_k, and this results in an additional exchange of $4K\phi$ bits. In Table 4, we show the amount of data (in bits) exchanged by the three entities in a round of execution of the protocol.

From these considerations, it results that the additional time overhead T_{op} is given by the following formula:

$$T_{op} = N \log N \cdot \tau_{eq} + \left(\frac{N!}{(N-2)!2!} + N\right) \cdot \tau_{comp} \quad (5)$$

where τ_{eq} and τ_{comp} refer to the time required to perform an equality and a comparison operation, respectively.

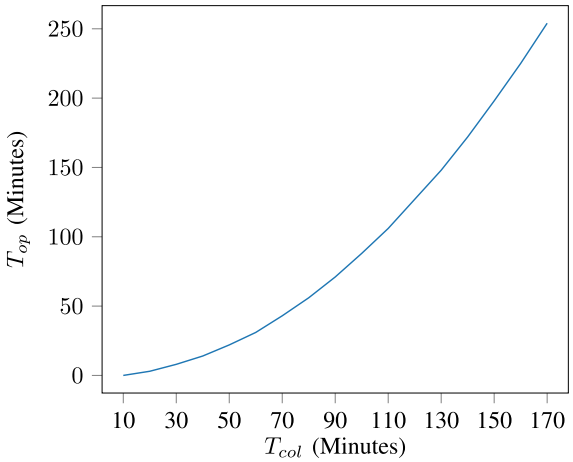


FIGURE 9. Time needed to perform the operations on the shares vs Period of collection of the shares (an arrival rate $\lambda = 1$ *req/s* is considered).

By discarding the operations that can be performed offline, we obtained $\tau_{eq} \simeq 0.47$ *ms* and $\tau_{comp} = 0.68$ *ms* on a Intel Core I7 computer. A representation of the time overhead needed to perform operations on the shares (i.e., T_{op}) as a function of T_{col} is depicted in Fig. 9.

Concerning the overhead introduced by the execution of the protocol, the volume of data exchanged with the CPs can be considered negligible. Conversely, the overhead of data exchanged between ISP and RA grows quadratically with the number of requests issued during the collection phase and with the bit-length representation of the shares (i.e., ϕ). With $\phi = 13$ bits, it is possible to generate unique shares during a collection phase that lasts up to 135 minutes, considering an arrival rate of 1 *req/s*. With these parameters, we obtain an overhead of $\simeq$ 2.2Gb online and $\simeq$ 5Mb offline considering $T_{col} = 80$ minutes. This overhead drops to $\simeq$ 1.2Gb when $T_{col} = 60$ minutes and to $\simeq$ 138Mb when $T_{col} = 20$ minutes. This overhead is acceptable, especially considering the traffic reduction achievable by the ISP. Remarkably, low values of T_{col} does not only guarantee the lowest data overhead, but also the lowest RO (as results from the analysis described in Section VII-B.1). Moreover, the negative impact of such overhead may be further reduced by colocating the RA with the ISP (e.g., as a virtual machine). We stress on the fact that the popularity-driven subdivision needs to be computed for each cache storage. In this work, we consider that the storage size can be of two types only: capacity of metro-core nodes and capacity of metro-aggregation nodes (i.e., 10% and 5% of the total size of the CPs' catalogues, respectively). Hence, the considered overhead needs to be accounted twice. Notice, however, that this overhead is still acceptable, and it would be acceptable even if more possibile caches' capacity was available. For example, if 10 types of cache sizes were present, it would be required to execute the protocol 10 times. This would imply, considering $T_{col} = 60$ minutes, an overhead of 12 Gb, which is $\sim$ 4 times the average size of the CPs' contents in our simulations.

VIII. CONCLUSION

In this paper, we proposed a privacy-preserving network-neutrality-compliant protocol for caching of VoD contents in ISP networks. The protocol guarantees that the ISP assigns portions of its caches' storage to several CPs proportionally to the popularity of their contents (i.e., *popularity-driven* subdivision) and it is therefore compliant with neutrality requirements recently proposed in the literature. Besides ensuring a NN-compliant caching, the protocol also allows to meet CPs' and ISP's privacy requirements, as the information about contents' popularity and size of cache are not disclosed. We evaluated how caching performance is influenced by a popularity-driven-subdivision in terms of overall network resource occupation and hit-ratio for ISP and CPs comparing it to baseline approaches, namely, *static subdivision*, where CPs are assigned the same amount of storage independent of their popularity, and *resource-occupation-driven*, where CPs are assigned an amount of storage according to amount of capacity their requests occupy in ISP's network. To this aim, we developed a dynamic VoD content caching and distribution simulator. We found that the popularity-driven and the resource-occupation-driven subdivisions lead to a reduction of the RO of up to $\sim$ 52% (and to an improvement of the hit-ratio of up to $\sim$ 32%) with respect to the static subdivision. In particular, the minimum RO and the maximum hit-ratio are obtained with the popularity-driven subdivision computed with our protocol. Moreover, we observed that the RO is highly-influenced by the frequency of execution of the protocol, that we assume to be tuned by the ISP in order to minimize the RO. Numerical results show that each CP experiences a loss in terms of its hit-ratio with respect to the case where it could selflishly establish this frequency. In the considered scenarios, this loss can range from a minimum of 0% to a maximum of $\sim$ 27% and it is much less significant when CPs' popularity are similar. Overall, our protocol proved to be beneficial in increasing caching performance (e.g., RO is reduced) while ensuring the protection of privacy. Note that privacy is protected also using the benchmark approaches, but none of them guarantees that the subdivision is actually compliant with NN requirements (as our protocol, instead, ensures). We also evaluated the data overhead introduced by the protocol and we conclude that it is acceptable compared to the reduction of RO experienced by the ISP. As a future work, we plan to extend our study considering more challanging security models (e.g., malicious parties that can alter their data during the execution of the protocol).

ACKNOWLEDGMENT

Davide Andreoletti and Silvia Giordano are funded by the Swiss National Science Foundation (SNSF) via the CHIST-ERA project UPRISE-IoT. Giacomo Verticale's work has been supported by the project ATMOSPHERE, funded by the Brazilian Ministry of Science, Technology and Innovation (Project 51119-MCTI/RNP 4th Coordinated Call) and by the European Commission under the Cooperation Programme, Horizon 2020.

REFERENCES

[1] "Cisco visual networking index: Forecast and trends, 2017–2022," Cisco, San Jose, CA, USA, White Paper, 2018, vol. 1.
[2] J. Krämer, L. Wiewiorra, and C. Weinhardt, "Net neutrality: A progress report," *Telecommun. Policy*, vol. 37, no. 9, pp. 794–813, 2013.
[3] P. Maillé, G. Simon, and B. Tuffin, "Toward a net neutrality debate that conforms to the 2010s," *IEEE Commun. Mag.*, vol. 54, no. 3, pp. 94–99, Mar. 2016.
[4] D. Andreoletti, S. Giordano, C. Rottondi, M. Tornatore, and G. Verticale, "To be neutral or not neutral? The in-network caching dilemma," *IEEE Internet Comput.*, vol. 22, no. 6, pp. 18–26, Nov./Dec. 2018.
[5] D. Andreoletti, C. Rottondi, S. Giordano, G. Verticale, and M. Tornatore, "An open privacy-preserving and scalable protocol for a network-neutrality compliant caching," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[6] B. Van Schewick, "Towards an economic framework for network neutrality regulation," *J. Telecommun. High Technol. Law*, vol. 5, p. 329, 2006.
[7] T. Garrett, L. E. Setenareski, L. M. Peres, L. C. E. Bona, and E. P. Duarte, "Monitoring network neutrality: A survey on traffic differentiation detection," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2486–2517, 3rd Quart., 2018.
[8] C. Pappas, K. Argyraki, S. Bechtold, and A. Perrig, "Transparency instead of neutrality," in *Proc. 14th ACM Workshop Hot Topics Netw.*, 2015, p. 22.
[9] X. Yuan, X. Wang, J. Wang, Y. Chu, C. Wang, J. Wang, M.-J. Montpetit, and S. Liu, "Enabling secure and efficient video delivery through encrypted in-network caching," *IEEE J. Sel. Area Commun.*, vol. 34, no. 8, pp. 2077–2090, Aug. 2016.
[10] D. Andreoletti, O. Ayoub, S. Giordano, G. Verticale, and M. Tornatore, "Privacy-preserving caching in ISP networks," in *Proc. IEEE 20th Int. Conf. High Perform. Switching Routing (HPSR)*, May 2019, pp. 1–6.
[11] A. Araldo, G. Dán, and D. Rossi, "Caching encrypted content via stochastic cache partitioning," *IEEE/ACM Trans. Netw.*, vol. 26, no. 1, pp. 548–561, Feb. 2018.
[12] D. Andreoletti, S. Giordano, G. Verticale, and M. Tornatore, "Discovering the geographic distribution of live videos' users: A privacy-preserving approach," in *Proc. IEEE Global Commun. Conf.*, Dec. 2018, pp. 1–6.
[13] K. V. Jónsson, G. Kreitz, and M. Uddin, "Secure multi-party sorting and applications," *IACR Cryptol. ePrint Arch.*, vol. 2011, p. 122, 2011.
[14] T. Turban, "A secure multi-party computation protocol suite inspired by shamir's secret sharing scheme," M.S. thesis, Dept. Telematics, Fac. Inf. Technol. and Elect. Eng., Norwegian Univ. Sci. Technol., Trondheim, Norway, 2014.
[15] A. Beimel, "Secret-sharing schemes: A survey," in *Proc. Int. Conf. Coding Cryptol.* Springer, 2011, pp. 11–46.
[16] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, Springer, 1999, pp. 223–238.
[17] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
[18] D. Beaver, "Efficient multiparty protocols using circuit randomization," in *Proc. Annu. Int. Cryptol. Conf.*, Berlin, Germany, Springer, 1991, pp. 420–432.
[19] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Proc. Annu. Cryptol. Conf.*, Berlin, Germany, Springer, 2012, pp. 643–662.
[20] O. Ayoub, F. Musumeci, D. Andreoletti, M. Mussini, M. Tornatore, and A. Pattavina, "Optimal cache deployment for video-an-demand delivery in optical metro-area networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.
[21] Q. Chen, S. Shi, X. Li, C. Qian, and S. Zhong, "SDN-based privacy preserving cross domain routing," *IEEE Trans. Dependable Secure Comput.*, to be published.
[22] O. Ayoub, F. Musumeci, M. Tornatore, and A. Pattavina, "Techno-economic evaluation of CDN deployments in metropolitan area networks," in *Proc. Int. Conf. Netw. Netw. Appl. (NaNA)*, 2017, pp. 314–319.
[23] J. Erman, A. Gerber, K. K. Ramadrishnan, S. Sen, and O. Spatscheck, "Over the top video: The gorilla in cellular networks," in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf.*, 2011, pp. 127–136.
[24] S.-E. Elayoubi and J. Roberts, "Performance and cost effectiveness of caching in mobile access networks," in *Proc. 2nd Int. Conf. Inf.-Centric Netw.*, 2015, pp. 79–88.
[25] M. Mangili, F. Martignon, and A. Capone, "Performance analysis of Content-Centric and Content-Delivery networks with evolving object popularity," *Comput. Netw.*, vol. 94, pp. 80–98, Jan. 2016.